Security Lab – Hacking-Lab Introduction and Setup

1 Introduction

The Hacking-Lab (HL) is an online ethical hacking and security challenge platform. Such infrastructures are also called Cyber Ranges. In its core, the Hacking-Lab provides access to (vulnerable) services, applications, and (virtual) devices to run attack- and defense exercises. There are two main ways how these resources can be accessed and how they are deployed.

Access:

- Directly over the Internet: In most cases, this access is used.
- In addition, over VPN: This is required for some exercises and gives you access to some Hacking-Lab-internal systems that are not exposed on the Internet.

Deployment:

- Personalized: You have your own version of the resource. This is typically done by deploying a Docker container within the Hacking-Lab and providing you with information (e.g., the URL) how to access it.
- Shared: You are sharing a resource with others, which usually means that others will be able to attack/use the resource as well, and at the same time.

The reason why such cyber ranges exist is that using hacking tools and doing attacks is unproblematic in these environments. It is a playground to do (almost) anything you want. The *Terms and Conditions*, to which you must agree when creating an account in the Hacking-Lab, tell you about the exceptions. The most important ones are described in Section 5 – *Rules of Online Conduct* and Section 12 – *User Solutions*. Section 12 states that you are not allowed to publish solutions or hints to the challenges in the Hacking-Lab.

2 Goal and Tasks

The goal of this lab is to prepare yourself so that you are ready to do exercises in the Hacking-Lab. To do this, work through the following tasks.

2.1 Create a Hacking-Lab Account

Each student needs a Hacking-Lab account. Create it as follows:

1. ZHAW has its own Hacking-Lab entry point/dashboard, which can be accessed at *https://zhaw.hacking-lab.com*
2. Select *Sign-in or Register with Global Hacking-Lab SSO*
3. Select *Register* and complete the registration. **As your username, use your ZHAW username in combination with -win or -zh**, depending on whether you are in the Winterthur (win) or Zurich (zh) class. Following this scheme allows the instructors to link the Hacking-Lab solutions you'll submit with your class and your lab team. For instance, if Michaela Stalder has the ZHAW username *staldmic* and is in the Winterthur class, use ***staldmic-win*** as Hacking Lab username. Conversely, in the Zurich class, ***staldmic-zh*** should be used. As e-mail address, use your ZHAW e-mail address.
4. Eventually, you should get access to the Hacking-Lab.

2.2 Get Access to the Hacking-Lab Exercises

First, you must get access to the SWS1 Hacking-Lab exercises. To do this, select ENTER ACCESS CODE at the top and enter the access code you get from your instructor. As a result of this, you should now see the SWS1 curriculum.

Note: You are not allowed to share this access code with anybody not enrolled in the SWS1 module.

2.3 Hacking-Lab Overview and Virtual Machine

The Hacking-Lab is organized in *curricula*, *events* and *challenges*. What you should see now is the curriculum *SWS1 – HS 2023*. Select this curriculum and you should see six events. You will work through all these events during this semester. Click on event *Introduction and Setup*, which shows you the four challenges that are relevant for now. Next, do the following steps:

1. Basically, you can access the Hacking-Lab using any system. However, the Hacking-Lab also provides a Hacking-Lab Virtual Machine (VM), which has the advantage that it includes all tools that are required to work on the challenges. Therefore, it's highly recommended to use the VM. There are different ways how you can use this VM, pick the one which works best for you:

   - Using the VM that is provided to you in the InIT Cloud. For additional details, refer to the document *Security Lab SWS1 – Using Virtual Machines in the InIT Cloud*, which you find on SWS1 Moodle in directory *Lab Information* under *Module Materials*. This is certainly the easiest way as everything is already set up and no local installation is needed.
   - Running the VM locally on INTEL-based computers. In this case, select the challenge *Kookarai Pentesting Linux: Installation INTEL Computers* and work through the challenge by following the step-by-step instruction. Note that the steps describe how to do this using VirtualBox. However, VMware Player and Fusion work as well (simply import the downloaded ova-file). Once you have completed the installation, check whether all tools are installed in the VM. To do this, select the *Applications* button (top-left) followed by *03 Web Application Analysis*, which should list several tools (e.g., *burpsuite*, *commix*, *skipfish*, *sqlmap*,…). If this is not the case, open a terminal and install the complete set of tools using *sudo apt install kali-linux-default* (followed by password *compass*).
   - Running the VM locally on Apple M1/M2-based computers. In this case, select the challenge *Kookarai Pentesting Linux: Installation Apple M1/M2* and work through the challenge by following the step-by-step instruction. Note that although the description uses *Parallels*, it is also possible to use VMware Fusion or VirtualBox to run the VM. If you are using a different virtualization software than Parallels, first download and install *Kali Linux for ARM64* (from *https://www.kali.org/get-kali/#kali-installer-images*) as a basis, select username *hacker* and password *compass* during installation, and before running *setup_parallels.ch* (you have to download the file before executing it), comment the following lines:

     ```
     # rename user parallels to user hacker
     #echo "* renaming user parallels to hacker"
     #sed -i -e 's/parallels/hacker/g' /etc/passwd
     #sed -i -e 's/parallels/hacker/g' /etc/shadow
     #sed -i -e 's/parallels/hacker/g' /etc/group

     # create /home/hacker dir
     #echo "* create empty /home/hacker directory"
     #mkdir /home/hacker
     #chown -R hacker:hacker /home/hacker
     ```

     When running the script, you'll eventually be asked whether the user parallels should be renamed. Although you already selected username hacker before, you must enter *yes* here. This is safe as the comments above make sure that nothing is actually changed.

2. Every challenge, when solving it correctly, gives you some *Hacking-Lab Points* – even the setup «challenges» you are currently using. Let's assume you used the challenge *Kookarai Pentesting Linux: Installation INTEL Computers* to install the VM locally on an INTEL-based computer. To get the points for this challenge, you simply have to submit the flag that is listed at the bottom of the step-by-step introduction. Enter it by clicking on *SUBMIT SOLUTION* near the top. If the correct flag is submitted, the system will automatically grant you the points. However, in most «real» challenges that you'll do during the semester, you must provide a written answer, which will be

graded by the instructor. If the answer is correct, you'll get Hacking-Lab points, which will translate to lab points in the SWS1 module. Note that you won't get any SWS1 lab points from the Hacking-Lab Points you get from the current setup «challenges», so submitting the solutions (the flags) merely serves to familiarize yourself with the solution submission system.

3. Next, in case you installed the VM locally, work through the challenge *Kookarai: Keyboard Layout* to set the keyboard layout according to your preferences. If you are using the VM in the InIT Cloud, you can ignore this step.

2.4 Burp and Firefox Web Developer Tools

Interceptor proxies are very valuable tools when analyzing and attacking web applications. The Hacking-Lab image provides two such tools, *Burp* and *ZAP*. Briefly familiarize yourself with *Burp* (as this will be the preferred tool in this module) by working through the corresponding challenge:

1. Select the challenge *Kookarai: Burp Inspection Proxy*. Make sure that you understand how to use Burp in principle. However, don't invest too much time for now to play around with it, because you'll do this anyway when working on later challenges. For now, it's mainly important that you know how to use Burp as a proxy in general and how to turn interception on and off. Note that if you also want to intercept responses with Burp (per default, only requests are intercepted), you have to activate this. To do so, go to the *Proxy* tab, click *Proxy settings*, go down to *Response interception rules*, mark the checkbox next to *Intercept responses based on the following rules* and in the table, additionally mark the checkbox of the row *Or … Request … Was intercepted*.
2. Another valuable feature you should know are the *Web Developer Tools* integrated in *Firefox*. There's no challenge in this context, but briefly try out these tools nevertheless. To activate them in Firefox, select menu *Tools* → *Browser Tools* → *Web Developer Tools*. The web developer tools offer various options to observe the network traffic (requests and responses), to inspect the browser storage (cookies, local storage, etc.), and more. Note that at the time of writing, there seems to be a bug in the sense that if elements in local/session storage or cookies are deleted or modified, then these elements are not always truly deleted or modified, even though the user interface may indicate this. To circumvent this bug, it is recommended to do the following (select the *Storage* tab first):
    - If you want to delete a cookie, right-click *Delete All Session Cookies* (instead of *Delete All*).
    - If you change the value of a cookie or of an element in local/session storage, always confirm this with the *Enter* key after you have entered the modified value.
    - Close and reopen the web developer tools after each deletion or modification, as this seems to make changes persistent and as this makes sure you see the current (updated) state of the cookies and of the local/session storage.
    - Likewise, if a cookie or an element in local/session storage has been set by the server-side but is not displayed, also close and reopen the web developer tools to make sure you see the current state.

2.5 Blocked Hacking-Lab Access and Workaround

If the Hacking-Lab environment gets too much network traffic from individual IP addresses or networks, source IP addresses or address ranges may be blocked temporarily (for about 10 minutes). When using the Hacking-Lab VM in the InIT Cloud, this is critical as all VMs access the Hacking-Lab from only a small number of source IP addresses. If a temporary blocking of IP addresses happens in this case, many or even all VMs will be affected and blocked.

To prevent such situations, don't use automated scanning tools (e.g., *nmap* or vulnerability scanners) against Hacking-Lab resources unless a challenge explicitly requires this.

In addition, in case you are blocked from the Hacking-Lab, accessing the Hacking-Lab over VPN may be a helpful workaround (use it only if problems occur). In this case, the VPN tunnel bypasses the firewall and provides direct access into the Hacking-Lab. To set up (and tear down) a VPN connection to

the Hacking-Lab, do the following in your VM (steps 2-4 are only needed once for some initial configurations, after that only steps 1 and 5-7 are required):

1. Use a terminal and navigate to */home/hacker/Desktop/OpenVPN*.
2. Edit *Hacking-Lab-2.0.ovpn* and change the IP address in line *remote 152.96.6.70* to *152.96.14.70* (the IP address of the VPN endpoint has changed in the meantime).
3. In the same file, add a line *mssfix 1300* below line *nobind* (this limits the maximum size of UDP packets used by openvpn to make sure they are smaller than the maximum packet size supported in the InIT Cloud).
4. Make the file *start_openvpn.sh* executable: *chmod 755 start_openvpn.sh*
5. If you are not logged in yet at *https://zhaw.hacking-lab.com,* use the browser to log in (the VPN tunnel can only be set up if you are currently logged in).
6. Execute *./start_openvpn.sh* to set up the tunnel. If the output ends with *Initialization Sequence Completed*, setting up the tunnel was successful. You can also use *ping nessus.vm.vuln.land* to check that the VPN tunnel works, as this host can only be pinged over VPN.
7. To tear down the tunnel, simply use *Ctrl-C* in the terminal. In case you have no longer access to the terminal where you set up the tunnel, use *sudo killall openvpn*.